

Appln No. 09/826,239

Amdt date April 15, 2005

Reply to Office action of November 19, 2004

REMARKS/ARGUMENTS

The above identified patent application has been amended and reconsideration and reexamination are hereby requested.

Claims 1 - 11 are now in the application. Claims 1 - 11 have been found by the Examiner to contain allowable subject matter.

The Examiner has objected to the pseudo-code listed in the specification on pages 48 - 53 as failing to comply with the writing requirement regarding the size of font and the position of the lines of code in the specification.

The Applicants have amended to specification to correct the size of font and location with the specification of the lines of code.

The Examiner has objected to Claim 9 for a misspelling informality. The Applicants have amended Claim 9 to correct the misspelling of the word "information".

The Examiner has raised the issue of inventorship of the claimed frame format subject matter and has asked the Applicants to provide a satisfactory showing that would lead to a reasonable conclusion that the applicants invented the subject matter disclosed in the cited article(s) and claimed in the application.

Accordingly, the Applicants submit the following:

In 1997 engineers at Epigram, Inc., who employed each of the inventors of the present application, began developing an approach for sending data throughout the home over phone lines.

In 1998 Epigram, Inc. and a number of other companies formed the Home Phone Networking Alliance (HPNA) as a business

Appln No. 09/826,239

Amdt date April 15, 2005

Reply to Office action of November 19, 2004

alliance for the purpose of promoting and standardizing home phone networking technology, and having a goal of preparing a technical specification that each of the members would adhere to.

In 1999 Epigram, Inc. in conjunction with Lucent Technologies submitted in confidence to the HPNA membership for consideration documents outlining the Epigram, Inc.-developed home networking system's technical details, which included the claimed inventive signal and method of transmitting, and in particular, the frame format, of the present application. Lucent Technologies did not contribute to any part of the development of the claimed inventive signal and method of transmitting, nor, in particular, the frame format, as set forth in the present application. The Epigram, Inc. / Lucent submission formed the basis of the resulting 1999 HPNA 2.0 Specification ratified by the HPNA membership.

Subsequent to the technology submission, inventors, Frank and Holloway submitted to the IEEE for publication in 2000 an article "Connecting The Home With A Phone Line Network Chip Set".

Further, the Applicants submit that the article "Quality of Support and Priority Management in Home PNA 2.0 Link Layer", while disclosing the frame format in its Figure 1 and referencing HomePNA 2.0 Specification dated Dec. 1999, appears to be merely a scholastic university engineering department article which analyzed in 2003 the Quality of Support mechanisms set forth in the 1999 HPNA 2.0 Specification. The authors Loh

Appln No. 09/826,239

Amdt date April 15, 2005

Reply to Office action of November 19, 2004

and Ozturk were not part of the technology development or Specification preparation team.

Therefore, in view of the above amendment and remarks it is submitted that the claims are patentably distinct over the prior art and that all the objections and concerns raised by the Examiner have been overcome. As such, issuance of a Notice of Allowance of the above Application is requested.

Respectfully submitted,

CHRISTIE, PARKER & HALE, LLP

By Richard J. Paciulan

Richard J. Paciulan

Reg. No. 28,248

626/795-9900

RJP/cah

CAH PAS596753.1-*--04/15/05 3:29 PM

equivalent transmission medium could be used to implement the present invention. For example, the transmission medium for the frame-based communications network could include power lines, or even wireless mediums, interconnecting transmitting and receiving stations.

(Deference:

Loop, looking for carrier sense, and when found determine whether the transmission was a collision or valid frame.

10 If it was a collision, process the signal slots and run the collision resolution algorithm.

In any case, then process the priority slots, looking for carrier.

15 Note that the "current" priority level is sticky from the slot the last collision occurred in.

Note that the Backoff Level (BL) and Maximum Backoff Level (MBL) counters are saturating at 0 and 15.}

Const

nPriorities = 8; {Number of priority levels}

20 nSignals = 3; {Number of signal slots}

nLevels = 16; {Number of Backoff Levels}

process Deference;

begin

25 currentPriority := 0; {Priority of the slot we are in}

cycle {deference loop}

sawFrame := false;

sawCollision := false;

while not carrierSense() do nothing; {watch for carrier

30 to appear}

deferring := true;

startTime := time();

stopTime := startTime;

```

        while carrierSense() do
            stopTime := time();
            if ((stopTime - startTime > CD_MIN) and
                (stopTime - startTime < CD_THRESHOLD)) or
5 collisionSense()
                then sawCollision := true
            else sawFrame := true;
            (After a collision, process the three signal slots)
            if sawCollision then
10 begin
                {wait until the end of the IFG, timing from start
of fragment
                reduces skew, since start-of-carrier uncertainty
is less than
15 end-of-carrier uncertainty }
                while (time() - startTime < CS_IFG + CD_FRAG) do
nothing();

                computeSignals();
                for (i := 0; i < nSignals; i++)
20 begin
                    startTime := time();
                    signal[i] := 0;
                    if signalSlot = i then sendSignal();
                    while (time() - startTime < SIG_SLOT) do
25 if carrierSense() then signal[i] := 1;
                end;
                processSignals();
                end;
                if (not sawCollision) then
30 begin
                    {wait until the end of the IFG}
                    while (time() - stopTime < CS_IFG) do nothing();
                    {If last transmission was successful, drop

```

```

Backoff Levels}
        BL[currentPriority] := saturate(0,nLevels-
1,BL[currentPriority]-1);
        MBL[currentPriority] := saturate(0,nlevels-
5 1,MBL[currentPriority]-1);
        end;
        {avoid timing hazard with transmitter,
currentPriority must be setup
        before deferring is cleared}
10        currentPriority := nPriorities-1;
        deferring := false;
        (Now time out the Priority (contention) slots)
        for (i := nPriorities-1; i>=0; i--)
        begin
15        slotTime := time();
        currentPriority := i;
        while (time()-slotTime < PRI_SLOT) do
            if carrierSense() then endcycle;{restart deference
loop}
20        {if priority slot passed with no contenders, then
that priority
            level must be idle, good practice says make sure the
backoff
            counters are reset}
25        BL[currentPriority] := 0;
        MBL[currentPriority] := 0;
        end;
        end; {cycle}
        end; {Deference}
30 {computeSignals: Determine which signals to send}
        function computeSignals();
        begin
            signalSlot := -1; {-1 means no signal to send,

```

```

initialization}
    if (txReady and (txPriority = currentPriority) and
BL[txPriority]=0) then
        signalSlot = integerRandom(nSignals); {select Backoff
5  Signal slot}
    end; {computeSignals}

{processSignals: Process the received signals, adjusting the
Backoff Levels}
10 function processSignals();
begin
    psignals := 0;
    for (i=0; i < nSignals; i++)
        if signal[i] then psignals++;
15 if (txReady and (txPriority = currentPriority)) then
    begin
        backoffLevel := BL[currentPriority];
        if backoffLevel = 0 then
            begin
20         tem := 0;
            for (i=0; i < signalSlot; i++)
                if signal[i] then tem++;
                BL[currentPriority] := saturate(0,nLevels-1,tem);
            end;
25         if backoffLevel > 0 then
            if psignals > 0 then
                BL[currentPriority] :=
                    saturate(0,nLevels-1,backoffLevel + psignals-1);
            end;
30 if psignals > 0 then
    begin
        if MBL[currentPriority] = 0 then MBL[currentPriority]
:= psignals;

```

```

        else    MBL[currentPriority]    =    saturate(0,nLevels-
1,MBL[currentPriority]
                                                    + psignals-1);

        end;
5  end; {processSignals}

{Transmitter:  Wait for txReady and txPriority from the link
level process.
  send txFinished when frame has been sent.}
10 process Transmitter;
begin
    cycle
        while (not txReady) do nothing();
        BL[txPriority] := MBL[txPriority];
15        while (not (txPriority >= currentPriority and
BL[txPriority]=0)
            or deferring)
            do nothing();
            ttime := time();
20            xmtDataOn(); {start data transmitting}
            while xmtBusy() and (time() - ttime < CD_FRAG) do
                begin
                    if collisionSense() then
                        begin
25                            xmtDataOff();{turn off, after sending minimum
collision fragment}
                                Ncollisions++; {timeout on excessive collision
limit}

                                    if Ncollisions = attemptLimit-1 then txFinished();
30                            endcycle;
                        end;
                    end;
                    while xmtBusy() do nothing();

```



```

        txReady := false;
        txFinished();      {signal link level that frame has
been transmitted}
        end; { cycle }
5  end; { Transmitter }

{collisionSense: }
function collisionSense();
begin
10      { When transmitting, detect the presence of a second
transmission.
        When receiving, detect overlapped transmissions}
end; { collisionSense }
{Receiver: }
15 process Receiver;
begin
    { Wait for carrier sense. Demodulate received signals into
frames.
    Reject collision fragments. Determine frame boundaries.
20 Check FCS.
    Filter based on destination address. Perform optional Link
Layer
    signaling and other controller functions.}
end; { Receiver }
25

```

RJP/cah

CAH PAS618378.1--04/15/05 11:59 AM